

WFQ 权重比较电路的设计

张 凡, 张盛兵, 黄小平

(西北工业大学 航空微电子中心, 陕西 西安 710065)

摘 要: 讨论了一种基于前导零的 WFQ 权重比较电路的设计, 主要描述了逻辑电路的设计方法、仿真和综合等。和传统的设计方法相比, 基于前导零方法的设计不仅具有速度的优势, 而且在电路的硬件资源消耗方面也具有较好的表现。

关键词: 前导零; 加权公平队列; 仿真; 可编程门阵列

中图分类号: TN4

文献标识码: A

文章编号: 1000-7180(2010)10-0027-04

The Design of Weight Comparator Circuit in WFQ

ZHANG Fan, ZHANG Sheng-bing, HUANG Xiao-ping

(Aviation Microelectronic Center, Northwestern Polytechnical University, Xi'an 710065, China)

Abstract: This article discusses a method about the design of Weight Comparator Circuit in WFQ based on leading zero. It mainly describes the method of design, simulation as well as the synthesis of the logic circuit. The result shows that the design based on leading zero not only has a speed advantage, but also has a good performance on hardware resource consumption.

Key words: leading zero; WFQ; simulation; FPGA

1 引言

随着 Internet 规模的不断增大, 各种各样的网络服务争相涌现, 如何在 Internet 中支持具有不同服务质量(QoS)要求的应用成为当前数据通信领域的研究热点^[1]。于是, 各种 QoS 技术应运而生。完善的队列调度算法是不同优先级队列中的数据获得所需服务质量的保证。在众多调度算法中, 加权公平队列(Weighted Fair Queuing, WFQ)的实现最复杂, 效果最好^[2]。

WFQ 是一种典型的常用于各种宽带和资源分配的调度算法, 常常用于队列数目不多但是各个队列的 QoS 要求很高的场合, 如 TSR \ GSR 骨干路由芯片的设计。相对于其他调度算法, 比如, 先进先出队列(FIFO)、轮转算法(Round Robin)、加权算法(Weighted Round Robin, WRR)等, WFQ 在调度的公平性、时延、时延抖动等方面都有更加的性能。因

此, WFQ 算法得到了广泛的应用。

2 WFQ 调度算法原理简介

WFQ 调度算法是一种基于时间戳(Timestamp)的调度算法^[3], 其基本思想是, 每个报文在入队时都被打上一个时间标记, 该时间标记表示报文的离开时间(Departure Time, DT), 出队时对 DT 进行比较, 选择 DT 值最小的报文出队。

假设一共有 16 个队列, 它们被分为 0~31 共 32 个优先级, 每个优先级队列可以单独配置权重。根据权重, 在这 16 个队列之间进行 WFQ 调度, 从而使它们按照配置比例共享某一端口资源, 调度过程简单描述步骤如下:

(1) 比较各个队列的权重值, 选择权重值最小的队列进行调度, 若权重值相同则选择队列号较小者;

(2) 所有队列的权重值都减去(1)中被选中的

收稿日期: 2009-07-20; 修回日期: 2009-09-20

基金项目: 国家自然科学基金(60773223)

- 队列的权重值,若结果小于零则取零值;
- (3) 在(1)中被选中的队列的权重值恢复到初始权重值至此完成一轮调度;
- (4) 重复以上三个步骤;

表 1 所示是一个四输入 WFQ 调度算法的调度过程演示(深色的队列号表示本轮被选中参与调度),此处没有考虑队列长度,仅参考配置权重值.

表 1 WFQ 调度过程演示

队列号	队列 0	队列 1	队列 2	队列 3
配置权重	3	5	2	5
	3	5	2	6
	1	3	2	4
	3	2	1	3
发生时标(动态权重)	2	1	2	2
变化过程	1	5	1	1
	3	4	0	0
	3	4	2	0
	3	4	2	6

通过上面的表格我们可以发现,队列获得调度的机会与它们的权重值成反比.当所有队列的配置权重都设置为 0 时,WFQ 实际上就是严格的顺序优先级调度算法,即队列号越小,优先级越高.

3 权重比较电路的设计实现

在实际应用中,基于性能的考虑,有时需要采用硬件实现 WFQ 调度算法,其中比较各个队列的权重值,选择权重值最小的队列进行调度,若权重值相同则选择队列较小者,这一步消耗的硬件资源较多,它需要在规定的时间内在多个数据中快速选择一个最小者.本文将介绍一种基于前导零的方法来实现设计 WFQ 调度算法中的权重比较模块的硬件电路.

3.1 设计方案

权重比较电路是 WFQ 硬件实现中的关键模块,它的资源消耗和硬件速度直接影响着 WFQ 调度算法的整体性能.因此,该部分电路既要保证速度足够快,又要保证电路的面积尽量小.

权重比较电路的设计方法很多,这里以 16 队列输入 WFQ 的调度为例,常见的设计思路如下:

首先,实现如图 1 所示的符合 WFQ 调度算法的二输入权重比较逻辑,输入包括两个队列的队列号 R_A[3:0]、R_B[3:0],权值 A_in[4:0]、B_in[4:0]以及两个队列的调度有效信号 Valid[1:0],输

出结果是符合 WFQ 调度算法的队列号、权值及其有效信号;然后,对于 16 队列输入的比较器则可以通过不断叠加使用上述二输入比较器来完成 16 组队列的权值比较任务,比较典型的做法是通过 16~8~4~2~1 逐级比较的 5 级流水来输出比较结果.这种方法的优点是模块复用度高,所有的五级比较电路都可以通过对二选一逻辑进行复用来实现,因此这种方法设计简单、容易实现;缺点是过多的流水线会造成较大的 latency^[4],特别是对于有 16 个输入队列的比较电路, latency 则会有 5 个周期,随着输入队列的增多, latency 会越大,而流水的减少则会造成组合逻辑的延时剧增,从而造成电路速度严重下降.显然,这种方法无法较好的满足系统要求.

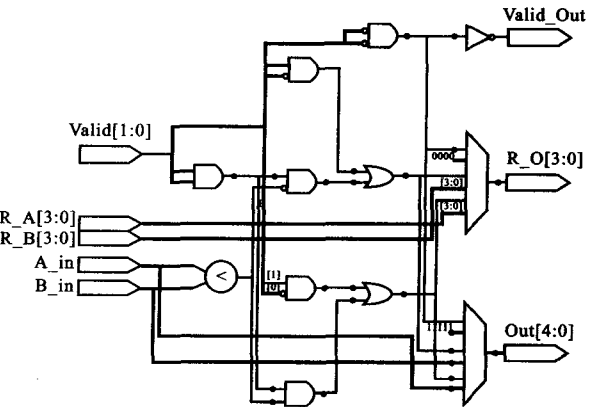


图 1 二输入权重比较逻辑

由于普通的基于电路叠加的设计方法无法较好满足系统在速度和面积这两点的要求,因此,本文设计了一种基于前导零的方法来实现权重比较电路.该电路的逻辑模块示意图如图 2 所示,共包括三大模块:预处理模块 M1,前导零计算模块 M2,结果处理模块 M3.首先,根据 WFQ 三种输入信号对各个队列的 5bit 权值进行 32bit 的加入 Valid 有效预处理的 One-hot 编码,然后对这 16 组预处理后的 One-hot 编码形式的权值进行按位或运算,就可以得到一个 32bit 的运算结果,通过对该结果进行前导零的计算,就可以得到 16 个队列中最小的权值.然后根据最小权值及队列的有效信息,经过比较就可以得出最小权值对应的队列号及其 Valid 有效信息.

3.2 预处理模块 M1 的设计

如图 3 所示,预处理模块 M1 包括两个子模块:第一个模块根据系统输入的 Valid 信号和配置权值 Data_In[0:15]进行 32bit 的 One-hot 编码,如果某队列的 Valid 无效,则该队列的权值 One-hot 编码全零,即为 32'b0,否则,该队列权值的十进制数对应

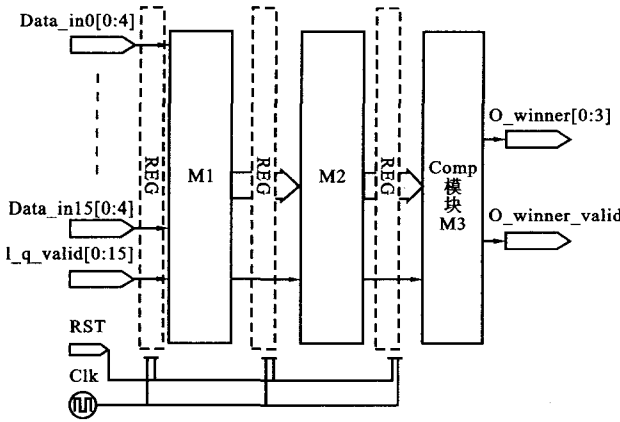


图2 前导零方法实现的权重比较电路

其 One-hot 编码前导零的个数。第二个模块的功能是对 16 个队列 One-hot 编码形式的权值进行按位或运算,产生的结果进行寄存器锁存后送入下一级模块 M2,即前导零计数模块 LZC 进行前导零计算。

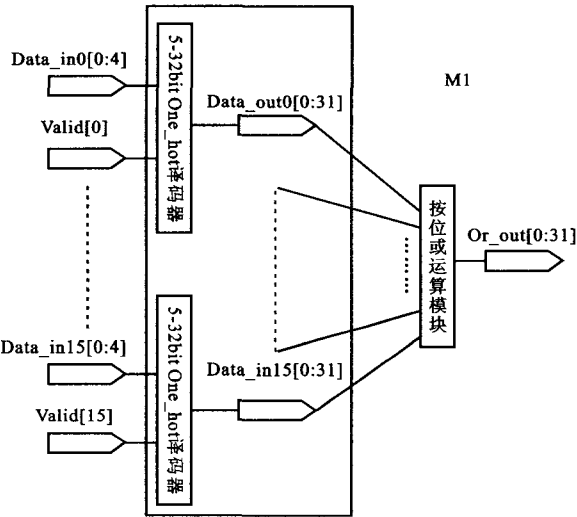


图3 预处理模块 M1

3.3 M2 前导零 LZC 的设计

LZC 模块负责对上一级产生的 32bit 运算结果进行前导零的计数,得出的前导零计数结果就是本轮 WFQ 调度的最小权值。该模块的实现如下:首先,实现一个如图 4 所示的 4bit 前导零计数器;然后以它为基础,利用 0 扩展(当某一块的所有输入全为零的时候输出位宽不足以表示这个数字,这时扩展位为零)的特性扩展出如图 5 所示的 8bit、16bit 直至设计所需的 32bit 的前导零。这样的优点在于扩展中充分考虑到零扩展的性质(当扩展位为 1'b1 时,表示这个模块可以精确表示前导零的个数,当为 1'b0 的时候,表示应扩展模块,此时边界处的值可以直接置出来),使整体设计的编码规整,结构清晰。

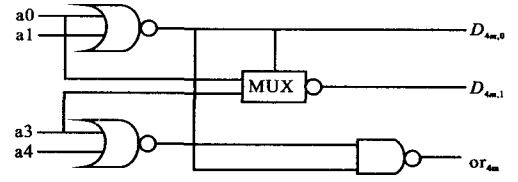


图4 4bit 前导零逻辑 lzd4

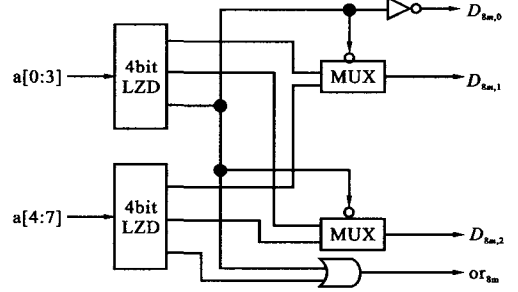


图5 8bit 前导零逻辑 lzd8

下面给出利用 0 扩展性来实现 8bit 前导零逻辑的源代码:

由两个 lzd4 模块构成一个 lzd8 模块:

```
//4bit 前导零 lzd4
lzd4
lzd4_u1(.a(a[0:3]), .d4m(d4m_1), .or4m(or4m_1));
lzd4
lzd4_u2(.a(a[4:7]), .d4m(d4m_2), .or4m(or4m_2));
//8bit 前倒零 lzd8 模块
//d8m:8bit 前导零输出
assign d8m[0] = ~ or4m_1;
assign d8m[1:2] = or4m_1 ? d4m_1 : d4m_2;
// or8m:8bit 零扩展端输出
assign or8m = or4m_1 | or4m_2;
```

3.4 结果处理模块 M3 的设计

上述的基于电路叠加的设计在整个电路比较过程中,每个队列的权值、队列号以及对应的 Valid 有效信号都是同步进入下一级比较器,进而得出最终结果。而基于前导零的设计在对 M1 的结果进行前导零计数时由于仅仅针对的是 One-hot 编码的权值,因此无法同步送出相应的队列号,所以这里需要一个额外的电路模块 M3。M3 模块的逻辑功能比较简单,它根据前导零模块得出的最小权值和系统输入的 Valid 有效值,并行的 16 组比较器来确定其对应的队列号,即得出最终的 WFQ 调度队列号,同时将结果送至寄存器输出,完成一轮 WFQ 调度算法的权值比较。

4 功能仿真和验证

功能仿真和时序验证对于检验设计的正确与否至关重要^[5]. 本文使用的仿真工具是 Mentor Graphics 公司的 ModelSim SE 5.8, 作者在详细分析 WFQ 调度特性的前提下, 有针对性地为逻辑电路的仿真编写了一个 Testbench, 然后通过对电路仿真的输出波形进行验证来判断电路逻辑功能是否正确^[6]. 仿真和验证的结果表明逻辑电路的调度结果和 WFQ 调度算法预期的调度结果完全保持一致. 因此, 我们可以得出结论: 本文设计的电路在逻辑功能上完全遵守 WFQ 调度算法的调度原理.

系统面积和时序的验证方面, 综合工具使用 Synplicity 公司的 Synplify Pro 8.1, 布局布线工具使用 Altera 公司的 Quartus II 7.1 (32-Bit). FPGA 验证芯片选择 Stratix II 系列的 EP2S15F484C3, 在不给设计添加任何时序约束的条件下, EDA 工具综合和布局布线的结果表明, 电路在时序上可以达到 451MHz, 相对采用电路叠加方法设计的电路有较大的速度优势; 资源占用方面, 电路共使用了 280 个 ALUT, 占 FPGA 芯片的资源的 2%; 寄存器共使用了 335 个 (One-hot 格式数据存储缺点就是占用内存较大), 占芯片总寄存器数的 3%, 在 FPGA 硬件资源不断增多的今天, 该电路的资源占用在可接受的范围之内^[7]. 可以看出, 本文设计的 WFQ 权值比较电路完全可满足系统在时序和面积方面的要求.

5 结束语

本文在对 WFQ 调度算法原理进行认真分析的

基础上, 提出一种基于前导零的 WFQ 权重比较电路的设计, 并对逻辑电路做了功能仿真、综合和布局布线. 通过仿真和验证的结果可以看出, 相对于传统电路叠加的设计方法, 基于前导零的电路设计在速度和资源占用方面有较大优势, 是一种值得采纳的问题解决方法.

参考文献:

- [1] 时公涛, 孟相如, 刘志宏. 基于 IP 网络服务质量的研究与实现[J]. 微电子学与计算机, 2005, 22(12): 116-119.
- [2] 杨超, 李士宁. 拥塞控制之 WFQ(加权公平队列)探讨[J]. 无线通信技术, 2003(1): 10-14.
- [3] 钟山, 岳祥. WFQ 流量调度算法研究[J]. 光通信研究, 2006(5): 16-18.
- [4] 夏宇闻. Verilog 数字系统设计教程[M]. 北京: 北京航空航天大学出版社, 2003.
- [5] 王钊, 卓兴旺. 基于 Verilog HDL 的数字系统应用设计[M]. 2 版. 北京: 国防工业出版社, 2007.
- [6] 科夫曼·K. 基于 Verilog 语言的实用 FPGA 设计[M]. 沈树群, 译. 北京: 科学出版社, 2004.
- [7] 徐光辉, 程东旭, 黄如, 等. 基于 FPGA 的嵌入式开发与应用[M]. 北京: 电子工业出版社, 2006.

作者简介:

张 凡 男, 硕士研究生. 研究方向为集成电路系统设计.

张盛兵 男, 教授. 研究方向为计算机体系结构和 VLSI 设计.

黄小平 男, 博士研究生. 研究方向为 ASIC 设计与验证.

(上接第 26 页)

作者简介:

姚放吾 男, 教授, 硕士生导师. 研究方向为嵌入式技术及其

在通信中的应用、并行计算机及其体系结构.

李晓辉 男, (1984-), 硕士研究生. 研究方向为并行计算机及体系结构.